Atypical Behavior Identification in large-scale Network Traffic

Daniel M. Best, Ryan P. Hafen, Bryan K. Olsen, William A. Pike

Abstract— Cyber analysts are faced with the daunting challenge of identifying exploits and threats within potentially billions of daily records of network traffic. Enterprise-wide cyber traffic involves hundreds of millions of distinct IP addresses and results in data sets ranging from terabytes to petabytes of raw data. Creating behavioral models and identifying trends based on those models requires data intensive architectures and techniques that can scale as data volume increases. Analysts need scalable visualization methods that foster interactive exploration of data and enable identification of behavioral anomalies. Developers must carefully consider application design, storage, processing, and display to provide usability and interactivity with large-scale data. We present an application that highlights atypical behavior in enterprise network flow records. This is accomplished by utilizing data intensive architectures to store the data, aggregation techniques to optimize data access, statistical techniques to characterize behavior, and a visual analytic environment to render the behavioral trends, highlight atypical activity, and allow for exploration.

Index Terms—Time series, large-scale data, visual analytics, cyber analytics.

1 Introduction

Advances in large-scale data collection infrastructures continue to outpace the human ability to process complex, heterogeneous data. Commodity processing power, storage, and pervasive sensing allow end users access to data volumes that promise deeper insight into complex phenomena. However, deeper insight can only be attained with effective interaction techniques to support knowledge discovery at scales exceeding what the visualization community has historically been prepared to deal with. Data volume and complexity pose challenges to fluid interaction with visualization tools, yet in many domains rapid interrogation of large data is necessary for critical event discovery and resolution.

The aim of Correlation Layers for Information Query and Exploration (CLIQUE) is to help network security analysts gain situational awareness in large, time-varying and potentially streaming data sets. Through behavioral summarization and anomaly detection techniques, CLIQUE provides insight into the nature of current activity on a network infrastructure through visual representations of typical and atypical patterns. CLIQUE is built upon a computationally low-cost statistical model, scalable data storage solution, and engaging visual analytic environment. Currently, CLIQUE development has been focused on the identification of atypical behavior in summary level computer network communication records, referred to as flows. These flow records are an abstraction of network traffic, aggregating individual packets into session-level summaries. Many enterprise network sensors process billions of network flow records per day. A single typical flow record is approximately 85 bytes to 250 bytes depending on summary metadata being stored about the communication. For that size of flow record, an enterprise recording 1 billion flow records per day would result in approximately 83GB to 244GB of uncompressed data per day. Despite this volume, summarizing packets as flows causes substantial loss of contextual information and content clues that help identify malicious events, complicating threat detection. Therefore, new techniques are needed to efficiently identify temporal patterns and potential threats within the massive amount of flow data. While flows are commonly used in network traffic visualization, many contemporary applications rely on visualization of raw flows and human perception alone to generate understanding of network behavior. CLIQUE provides a novel approach which introduces summary signals representing behavioral patterns in very large data sets.

This paper describes the three main components of CLIQUE that allow for the exploration at scale of behavior in network data: (1) the statistical model used to identify atypical behavior, (2) the data management considerations to allow for scalability, and (3) the visual analytic environment that allows users to explore behavioral trends interactively. We also reflect on the application's performance against the data volumes characteristically seen in operational environments.

- *Daniel M. Best is with Pacific Northwest National Laboratory, E-mail: daniel.best@pnl.gov.*
- *Ryan P. Hafen is with Pacific Northwest National Laboratory, E-mail: ryan.hafen@pnl.gov.*
- *Bryan K. Olsen is with Pacific Northwest National Laboratory, E-mail: bryan.olsen@pnl.gov.*
- *William A. Pike is with Pacific Northwest National Laboratory, E-mail: william.pike@pnl.gov.*

Manuscript received 31 March 2011; accepted 1 August 2011; posted online 23 October 2011; mailed on 14 October 2011.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

2 Related Work

Determining atypical behavior is important to analysts who deal with large volumes of time series data commonly experienced in cyber security, finance, power grid, and other domains [4]. There have been many applications in cyber security that try to visually represent the behavior of a network [11, 9, 18]. An introduction to some of these tools is provided by John Goodall in his "Introduction to Visualization for Computer Security" [8]. The need for a visual analytic tool to expose features of interest in large-scale datasets continues to rise as the amount of data to analyze increases.

Similar to the existence plots presented by Janies, we aim to summarize activity in a limited amount of space [11]. The existence plots allow for an analyst to quickly determine if there

is any atypical behavior that should be addressed. We improve upon this by plotting categories independently, therefore reducing overall complexity and overplotting. We also visually draw out significant deviation from normal activity using a gradient background and allow for aggregation of behavior into logical groupings of hosts.

VIAssist takes advantage of smart aggregation and coordinated views to ensure scalability and help analysts accomplish their tasks [9]. CLIQUE manages scalability through the use of aggregation as well, both by automatically storing aggregates in database tables, and by analyst defined group hierarchy. Instead of brushing and linking we utilize a time indicator on all cells based upon the position of the cursor relative in time to a particular cell.

Giving the analyst a holistic view of their information space enables them to see the bigger picture and make more informed decisions. VisAlert accomplishes this task by visualizing a graph of the network within a containing set of rings that represent time and type of alert [15]. An edge is then added from the alert to the host that generated the alert, giving users visual indication of misbehaving systems. We have taken a different approach to visualization using a grid that allows for resizing of rows and columns to show more items of interest while keeping data in context. However, the attributes of what (anomalous behavior), when (time indicator), and where (arbitrary grouping) are maintained.

There has been a significant amount of work associated with statistical anomaly identification for network intrusion detection systems. Very early work includes the IDES and NIDES systems [6, 12]. A summary of general anomaly detection methods are presented in [5], while state of the art anomaly detection methods specifically targeted at network systems are surveyed in [14]. In [17], it is argued that other than a few commercial network anomaly detection systems (e.g., [2], [3]), anomaly detection systems are virtually nonexistent in operational settings. They attribute this finding to the fact that identifying attacks in network data is a much more complex problem than in those found in other domains where statistical and machine learning techniques have been used successfully.

3 Statistical Methods

The statistical anomaly detection methods we have developed for CLIQUE are simple by design. Operating in a scalable, interactive manner on massive volumes of data is more attainable with a simple model. Although network activity behaviors may be very complex, our goal is to give the analyst insight into behaviors of interest present in their network. The goal is to optimize analyst efficiency by providing them with jump-off points warranting further in-depth investigation.

The model operates on the assumption that statistical patterns of time-aggregated enterprise network flow attributes exhibit cyclical behavior of a weekly periodicity. Exploratory analysis of a large volume of enterprise network traffic validated that this assumption holds quite well for most protocols. There are many other considerations, such as accounting for cyclical behavior with periodicity on the order of minutes to hours due to automated activity, more complex host-network interactions, etc. We have found that simply highlighting deviations of activity based on weekly periodicities to be very effective for drawing out events that analysts should further investigate.

3.1 Data

CLIQUE modeling is based on network flows, which are summaries of individual network connections, consisting of major variables such as time of origination, duration, protocol, source and destination port, packets, bytes, etc. These flow summaries may also contain a traffic category determined from a specified set of rules (such as web traffic, secure shell, etc.). The network flows can be aggregated into meaningful groupings, such as enterprise-wide, department-specific, or even a grouping important to the individual analyst. For a given network activity, category, and IP grouping, network flow data is aggregated over time and compared to a historical baseline to highlight atypical behavior. Throughout, the level of aggregation will be presented as one minute intervals, although this can and should vary based on properties of the data being processed. We treat one minute intervals as the smallest level of aggregation, and all results in this section would apply to higher levels of aggregation as well. The statistical validity of these methods relies on adequate aggregation. Although we present the method using counts of connections, other variables such as total bytes or total packets can be aggregated and compared.

3.2 Methods

Current data is compared to historical data on a minute-of-week basis. The model for comparing current counts to historical counts simply consists of calculating the mean behavior of the historical data and checking it against the current data. The assumption is that behavior for a given minute of the week in the past will persist in future weeks. With the frequently idiosyncratic behavior of network data, this is a very strong assumption, however it simplifies the modeling and calculations so they can meet the goal of interactivity, and tends to generally hold.

For a given category and IP grouping, let $x_1,\ldots,x_n$ denote n sequential observations for the current week. Suppose, for example, that we are monitoring data for a given Thursday from noon to 4:00 PM. Then $n = 240$ and x_1 would correspond to the count from 12:00 to 12:01, x_2 from 12:01 to 12:02, up to x_{240} corresponding to the count from 3:59 to 4:00. Let the historical observations from previous weeks corresponding to the current series be denoted as

$$x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}, \quad i = 1,\ldots,m.$$

Here, supposing we have m weeks of historical data to compare to, the superscript i corresponds to historical data occurring i weeks previously. So with the example above, $x_1^{(1)}$ would be the count from 12:00 to 12:01 the previous Thursday, etc. In the example provided in this paper, we use $m = 3$, although using a larger historical baseline would be advisable if resources are available.

Figure 1 shows an example of historical Network Time Protocol (NTP) data for a 240-minute period, showing the square root number of connections aggregated by minute for three weeks of data. We find very predictable aggregate behavior from one week to the next. The following sections will describe the steps of our statistical methodology using this data as an example, which consists of (1) characterizing the mean behavior of the historical data, (2) calculating the mean behavior of the current data, and (3) calculating a difference metric between the two.

3.2.1 Historical Data Mean

For a given time window $1,\ldots,n$, we summarize the past behavior using a running median across time. The median is used due

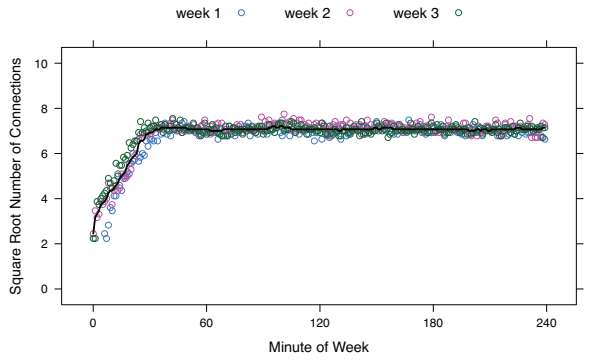

Fig. 1. Three weeks of historical data for NTP, from minutes 1 to 240, with fitted mean.

to its robustness to outliers. A running median consists of taking a block of k time windows and finding the median, then sliding that window across the time series. For example, if $k = 15$ and we wish to calculate the median at minute j, we would calculate the median of all the values corresponding to times $j-7$ to $j+7$. Near the endpoints, subsequently smaller medians are computed and Tukey's endpoint rule is used at the endpoints themselves [19]. Algorithms for the running median can perform as $O(n \log k)$ [10], although with k so small, brute force computation completes in reasonable time.

Ignoring endpoints for the moment, the historical mean using a running median with odd window width k at minute t is obtained as

$$h_t = \text{median}\left(x_j^{(i)} : i \in (1,\ldots,m), j \in \left(t - \frac{k-1}{2}, \ldots, t + \frac{k-1}{2}\right)\right)$$

The running median gives a good measure of the mean of the historical data and how it changes over time. We can then measure the variability of the historical data around this mean, which should give a good indication for the range of future values to be expected. It is assumed the variance of the counts around the mean is constant over time. This assumption does not hold very well when working with untransformed data, as higher counts will have larger variance. However, the square root and log transformations of the time-aggregated data seem to mitigate this issue.

We measure the variability using a robust measure, the median absolute deviation (MAD). The MAD is simply the median value of the absolute values of the deviations of the observed historical data $x_t^{(i)}$ from the mean, h_t. We will denote this deviation measure as $\hat{\sigma}_h$,

$$\hat{\sigma}_h = 1.4829 \text{ median}(|x_t^{(i)} - h_t| : i \in (1,\ldots,m), t \in (1,\ldots,n))$$

This variability estimate reflects both the variation within a given minute and across minutes.

The constant 1.4829 comes from the fact that if the MAD were calculated for a random sample from a normal distribution with unit standard deviation, $\sigma = 1$, then $\sigma \approx 1.4829\text{MAD}$. Thus, the constant puts this robust measure of variability on the same scale as the standard deviation of a normal random variable.

We can now construct bands around our historical data for which we expect future observations to fall within. On average we expect observations to be near the mean, give or take a certain number of standard deviations. For example, if the observations are normally distributed around the mean, we can expect 99.7% of the values to fall within 3 standard deviations from the mean. While the normal assumption might not hold strongly in all cases, we can use this as a rule of thumb, looking at bands constructed with multiples of 3+ standard deviations. Using a multiplier α, we can construct bands

$$h_t^{lower} = h_t - \alpha\hat{\sigma}_h$$
$$h_t^{upper} = h_t + \alpha\hat{\sigma}_h$$

The sensitivity of anomaly detection is based heavily on choice of α. Lower values of α would make the method more sensitive, and higher values could be chosen when only very outrageous behavior is desired to be seen.

Figure 2 shows the three weeks of historical NTP data with its fitted mean, h_t using a running median with $k = 15$, and a confidence band with $\alpha = 4$.

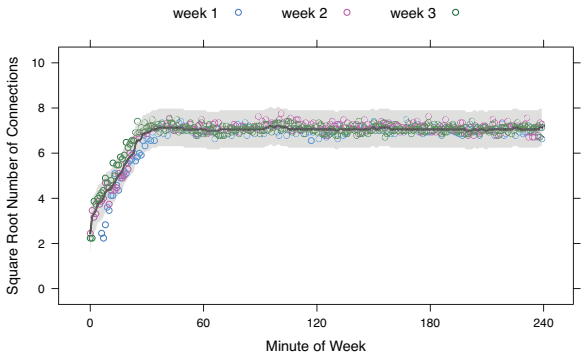

Fig. 2. Three weeks of historical data for NTP, with fitted mean with $k = 15$ and confidence bands with $\alpha = 4$.

3.2.2 Current

We take a similar approach to the current data series $x_1,\ldots,x_n$. The mean value over time is calculated using a running median,

$$c_t = \text{median}\left(x_j : j \in \left(t - \frac{k-1}{2}, \ldots, t + \frac{k-1}{2}\right)\right)$$

and using a measure of deviation

$$\hat{\sigma}_c = 1.4829 \text{ median}(|x_t - h_t| : t \in (1,\ldots,n))$$

we construct bands

$$c_t^{lower} = c_t - \alpha\hat{\sigma}_c$$
$$c_t^{upper} = c_t + \alpha\hat{\sigma}_c.$$

Figure 3 shows the current data series superimposed over the historical data, with the fitted mean and confidence bands. The current data series is very different from the historical series. In the following section we will discuss how to quantify and highlight these deviations.

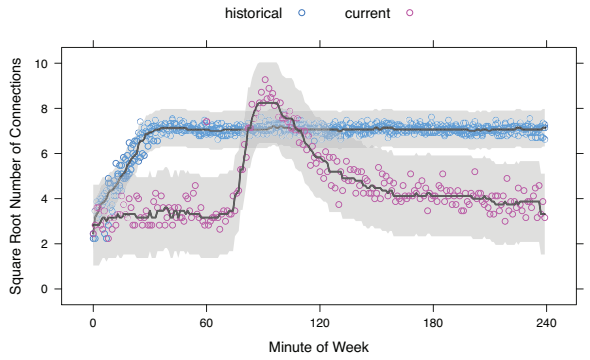

Fig. 3. Current and historical series for NTP data, with fitted means and bands.

3.3 Highlighting Atypical Behavior

To highlight atypical behavior in the current data series, we compare the current and historical series based on their mean and variance properties over time. Recall from our model that we are constrained by our assumption that counts of activity for a current minute-of-week behave similarly to counts of the same minute-of-week in the past. Thus, events found to be "atypical" by this model can simply be interpreted as being significantly different from counts seen in previous weeks.

To show how the means of the current and historical series differ, we display a difference of the means

$$\delta_t = c_t - h_t.$$

This gives a simple visual summary of how much the current series is deviating on average from the historical. Values of δ_t further from zero correspond to increasing atypical activity. To be able to quantify how far from zero is atypical enough to be worthy of attention, we need to take the variance into account.

If our current series falls in line with what has happened in the historical data, it should fall within the limits of the bands around the historical mean. When this is not the case, we want a metric to describe how different the current series is from the historical data. This metric is displayed in CLIQUE as a ramping of the color red from 0% to 100% saturation. We want 100% saturation to correspond to behavior in which the current series is completely out of range of the historical bands. We relax the saturation down to 0% as the historical and current bands begin to overlap, where 0% is reached when they completely overlap.

Figure 4 shows the original NTP data with the bottom part of the plot showing δ_t with shading highlighting significantly atypical behavior with the saturation calculated as described. From this, the analyst can determine whether further detailed investigation is warranted.

To put the saturation in mathematical terms, in cases where the historical mean is greater than the current mean, $h_t > c_t$, the overlap is calculated as

$$\lambda_t = \max(0, h_t^{lower} - c_t^{upper})$$

and if $h_t <= c_t$

$$\lambda_t = \max(0, c_t^{lower} - h_t^{upper})$$

Now, the saturation at time t is calculated as

$$s_t = \begin{cases} 0 & h_t - c_t \\ \min(1, \lambda_t / |\delta_t|) & h_t \neq c_t \end{cases}$$

Figure 5 shows s_t across time for the NTP data (compare to Figure 3).

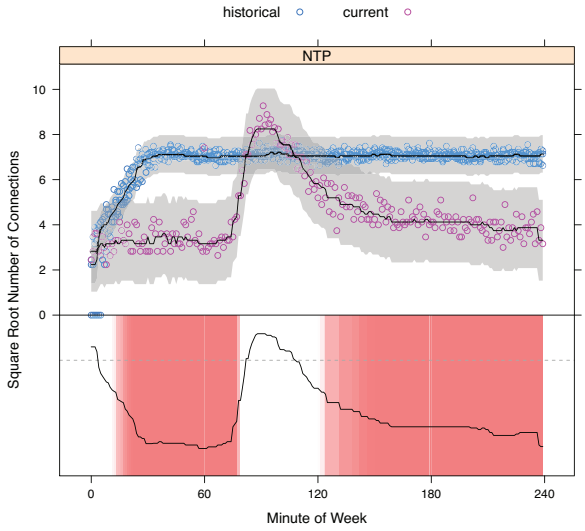

Fig. 4. NTP data with a difference chart and color representing deviation.

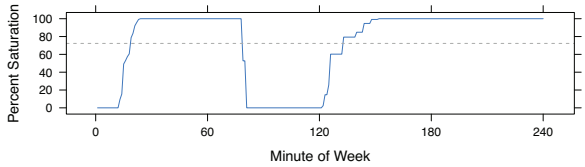

Fig. 5. Saturation level for NTP data.

3.4 Abrupt Outliers

The running median smooths out observations that abruptly depart from around the mean and immediately return back. In Figure 4, we see one such case around minute 60, where the current count is much higher than those around it (although in line with the baseline). This abrupt deviation is lost in the difference chart in the bottom panel of the plot. To bring attention to such points in the difference chart, we can draw a vertical line indicating these outlying points. For example, Figure 6 shows the same plot with a vertical line added for the outlying point.

To make identification of abrupt outliers automatic, we can flag any points within the current data series which, after being compared to the smoothed historical baseline, show a deviation beyond the regular deviation of the data. Specifically, we flag any point as being an outlier where $|x_t - h_t| > \alpha\sigma_c$. Currently this is not implemented in CLIQUE, however the capability has been added to the development path.

4 Data Management

It is important to leverage data intensive architectures when analyzing massive volumes of data typically found in enterprise perimeter network communication records. Parallel database technology has emerged as a scalable shared-nothing approach for storing massive structured datasets. We have leveraged

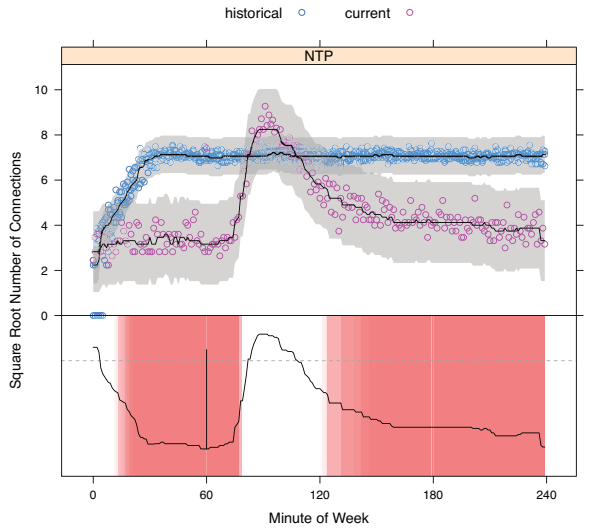

Fig. 6. NTP data with a difference chart and line indicating a single abrupt outlier.

a parallel and distributed approach by utilizing the Netezza TwinFin® 6 data warehouse appliance. The hardware environment consists of a host node with 8 Xeon 3.0 GHz CPU cores and 24GB of main memory. Also included are 6 S-Blades, each with 8 Xeon 2.6 GHz CPU cores and 16GB of main memory. Each CPU core in the S-Blades have a dedicated field-programmable gate array (FPGA), programmed to filter out extraneous data as fast as it streams off the disk. FPGAs reduce the I/O bottleneck and eliminate the processing of unnecessary data, improving overall system and query performance.

The Netezza TwinFin® system is built on a unique asymmetric massively parallel processing (AMPP™) architecture that combines open, blade-based servers and disk storage with data filtering using FPGAs. This combination delivers fast query performance and modular scalability [1]. Shared-nothing architectures offer the advantage of being able to scale as the size of data increases and minimize interference by minimizing resource sharing and data movement. Their main advantage is the ability to be scaled up to thousands of processors and offer near-linear speedup as the hardware is scaled up on complex relational queries [7].

The Netezza TwinFin® allowed us to distribute enterprise network perimeter traffic across all of the disks in the storage array, providing better query performance than a traditional relational database management system (RDBMS). We also determined that with this architecture it is possible to use a multi-threaded approach to query execution. CLIQUE determines how many processors are available on the client machine and calculates how many threads would be appropriate. This way, if a host can support many threads, they will be utilized, while older hardware will not be penalized for not having the capability. The client application computes behavioral deviations locally, meaning the database is only responsible for providing information for a relevant subset of data based on IP address ranges and categories of traffic. The application plots each cell in the matrix independently, so we can issue multiple queries against the database in parallel, as well as have the queries executed on the appliance in parallel. This implementation allowed for significant performance improvement to the overall load time of the interface.

4.1 Aggregation

The massive data volume produced by hundreds of millions to billions of network flows led to the exploration of alternative summary tables to make the application more load time efficient and interactive. We determined that summarization should be done on only relevant IP addresses, defined as internal to the organization. By limiting our entities of interest to only those internal to the organization, it enables characterization of behavior relevant to most organization analysts. When implementing summary or aggregate tables, it is critical to define the granularity of data that will be stored within the table. We determined that aggregating to the minute would facilitate data volume reduction, while still providing flexibility for an implementation to aggregate to a wider interval of time. It is also important to determine the measures which should be summarized for each interval of time. For this application, the count of communications within a specific interval for each entity of interest was calculated. This provides the ability to analyze typical traffic patterns within an interval over time. Since many applications categorize traffic, it was important to enable efficient summarization by category as well. The granularity of the aggregate table can be defined as a record describing a count of network communications for each IP address, for each minute of time, and for each category of traffic.

Listing 1 describes the summary table definition as implemented on Netezza. There are several methods to populate the summary table from the original network flow table. We decided to implement a database view which includes two queries combined using a UNION clause that also incorporates the CASE logic necessary for categorization of traffic. The first query selects all internal IP addresses that exist as a source IP address in the communication; the second query selects all internal IP address that exist as the destination address. The categorization is determined by key features in the data such as protocol, source port, destination port, packet count, payload, etc. For example, if the destination port is 80 or 8080 then the traffic is categorized as 'WEB' traffic. The rule-based approach allows analysts to add, remove, and refine rules as knowledge is gained. The summary table can then be populated incrementally by bounding the start and end time of the query when inserting data into the summary table from the view. This approach does not require stored procedures or functions which are typically written in database specific languages. A simple example of the insert is included in Listing 2 and is implemented by substituting start time and end time variables with actual string date values.

```
CREATE TABLE SUMMARY_IP
(
    IP              BIGINT          NOT NULL,
    CATEGORY        VARCHAR(13)     NOT NULL,
    INTERVAL_TIME   BIGINT          NOT NULL,
    INTERVAL_COUNT  INTEGER         NOT NULL,
)
DISTRIBUTE ON (IP)
ORGANIZE ON (CATEGORY, INTERVAL_TIME, IP);
```

Listing 1. Summary Table Definition

```
INSERT INTO summary_ip
    ( ip
    , category
    , interval_time
    , interval_count )
SELECT
      ipaddr
    , category
    , interval_time
    , interval_count
FROM  v_summary_ip
WHERE interval_time >=
      EXTRACT(EPOCH FROM start_time::DATETIME)
AND   interval_time <=
      EXTRACT(EPOCH FROM end_time::DATETIME);
```

Listing 2. Example Summary Table Insert

5 Visual Analytic Approach

In order to allow human cognition to understand the data, it is important to provide users with a usable environment to interact with the data [13]. Additionally, due to the volume of data, information visualization has been utilized by many to provide users the ability to get a bigger picture of the system. These two items and the ability to explore data to ask questions provide the user with a rich visual analytic environment. CLIQUE based its structure of interaction on LiveRac, an accordion drawer tool that allows for expanding and collapsing cells while the others remain visible [16].

CLIQUE uses several techniques to accommodate large amounts of data to provide analysts an environment for investigation. Hierarchical grouping provides the ability to logically group actors, while detail on demand reduces the amount of data needed to be presented at any one time. Finally, the difference plot based on the statistical model provides analysts insight on what groups or individual actors are acting atypically.

Fig. 7. CLIQUE interface.

Figure 7 shows an initial view of the interface. The hierarchy in this example is defined from the highest to lowest level of granularity beginning with site, facility, organization group, and finally individual IP address. The IP address groupings are shown along the left side and comprise the rows of the grid. Along the top, different groupings of traffic make up the columns. These groups can be defined in various ways from simple decision trees to clustering based on traffic type. Each row column intersection (cell) shows the traffic for the group for the given category.

5.1 Hierarchical Grouping

A difficulty with exploring large network traffic data is not only the amount of data, but also the number of distinct actors that are available for investigation. In a moderately sized network there can be millions of IP addresses an analyst could be interested in viewing. A common choice is to separate the records into hierarchical groups and allow for drill down to expose more actors. When going with this mechanism, decisions must be made about where to store the groupings, who has access to view the groups, and how the groups are created or updated.

To remain flexible, CLIQUE allows for arbitrary groupings defined in an XML structure stored on an individual analyst's hard drive. This enables an analyst the ability to customize the groups of interest for their purposes. Grouping could also be developed using the network traffic point of origin, or separated by the assumed role of the systems within the group. Giving the analyst control over specific groupings enables sharing with other analysts to verify atypical behavior.

Arbitrary groups work well in CLIQUE because the data is retrieved from the database quickly, and the statistical model developed allows for calculation of atypical behavior in real-time. Without those two capabilities CLIQUE would need to either pre-calculate behavior, store groups on the server, or possibly both. All would hamper the free flow of exploration for an analyst. Currently, groups are defined using a simple GUI that can generate a new file to explore within minutes by allowing analysts to enter groups hierarchically by Classless Inter-Domain Routing (CIDR) notation, individual IP addresses, or IP address range.

Interacting with the groups is as simple as double clicking on a group name to drill into the data, or clicking on a breadcrumb to navigate to any level in the hierarchy. This interaction ensures the user knows where they are, where they have been, and where they are going in the hierarchy. This context of location is important when understanding the actors involved in atypical behavior.

5.2 Detail on Demand

Detail on Demand allows for summary data to be presented and further data to be retrieved only when the analyst is interested in getting more detail about a particular group and category of traffic. By presenting the summary data detail on demand enables the interface to remain interactive by not retrieving or calculating more than is needed. Additionally, it helps reduce the likelihood of cognitive overload associated with presenting too much at the same time.

The hierarchical grouping discussed in Section 5.1 is one form of using the detail on demand method. If more data is desired about a given group, the analyst drills in to view a subgroup and can continue doing so until individual IP addresses are shown. The other interaction in CLIQUE that provides detail on demand is the ability to stretch open a given cell, a capability featured in LiveRac. While stretching the cell, additional plots can be generated based on the data, and if necessary, additional data can be retrieved from the database. This interaction ranges from when cells are too small to plot data to when they are given the majority of the visual space. When the cells are

small a heat map coloring is presented. At the other extreme a behavioral plot with several series of data is presented along the chart labels and legend.

5.3 Difference Plot

The difference plot takes the statistical model discussed in Section 3 and represents the data in the interface to highlight atypical behavior as discussed in Section 3.3. The difference plot requires the most data to render, therefore it is currently set as the lowest level of detail presented.

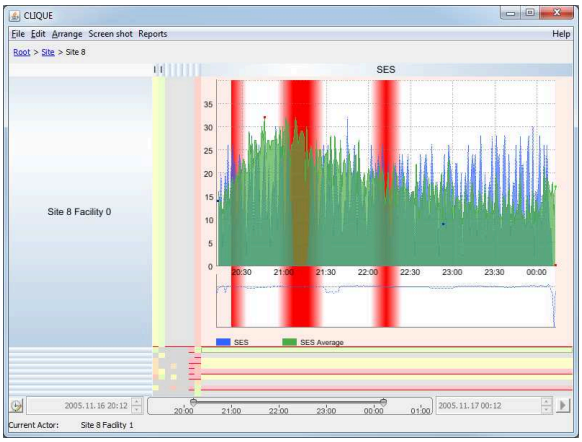

Fig. 8. Difference plot in CLIQUE showing atypical behavior for a given actor.

As seen in Figure 8, the atypical behavior is represented as a gradient background to the historic and current plots. Saturation of red is used to show the continuous scale from typical to highly atypical behavior. The top of the scale based on deviation is different for each chart, as the variability of the data is taken into account by the statistical model. The maximum deviation (100 percent saturation) is configurable to any standard deviation desired by the analysts. The default is 4 standard deviations as this represents fairly atypical behavior. If desired, an analyst can adjust this to focus only on highly atypical behavior or consider anything even remotely atypical as a cause for alarm.

6 Conclusion

In this paper we have presented an implementation of a method to identify and present atypical behavior to analysts. The statistical model provides the ability to highlight activity that deviates from normal for a given host. By using this model, it allows the client to use groupings of IP addresses formed into a hierarchy so that analysts can fine tune the information space they are interested in and at which level they want to view. The interface and model both rely on the scalability of the database to retrieve and aggregate data so that it can be presented. To achieve identification of atypical behavior in large-scale data sets, all three components (model, visualization, and storage) must be given adequate consideration.

Acknowledgments

This research was supported by the U.S. Department of Homeland Security Science and Technology Directorate. The authors are grateful to the analysts who shared requirements and evaluated prototypes for this work. The Pacific Northwest National Laboratory is managed for the U.S. Department of Energy by Battelle Memorial Institute under Contract DE-AC06-76RL01830.

References

[1] Netezza TwinFin® Data Sheet. http://www.netezza.com/documents/twinfin_ds.pdf.
[2] Peakflow. http://www.arbornetworks.com/en/peakflow-sp.html.
[3] Stealthwatch. http://www.lancope.com/products/.
[4] M. Cahill, D. Lambert, J. Pinheiro, and D. Sun. Detecting fraud in the real world. *Computing Reviews*, 45(7):447, 2004.
[5] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):1–58, 2009.
[6] D. Denning. An intrusion-detection model. *Software Engineering, IEEE Transactions on*, (2):222–232, 1987.
[7] D. DeWitt and J. Gray. Parallel database systems: the future of high performance database systems. *Commun. ACM*, 35:85–98, June 1992.
[8] J. Goodall. Introduction to Visualization for Computer Security. *VizSEC 2007*, pages 1–17, 2008.
[9] J. Goodall and M. Sowul. VIAssist: Visual analytics for cyber defense. In *Technologies for Homeland Security, 2009. HST'09. IEEE Conference on*, pages 143–150. IEEE, 2009.
[10] W. Härdle and W. Steiger. Optimal median smoothing. *Applied statistics*, 44(2):258–264, 1995.
[11] J. Janies. Existence plots: A low-resolution time series for port behavior analysis. *Visualization for Computer Security*, i:161–168, 2008.
[12] H. Javitz, A. Valdes, and C. NRaD. The NIDES statistical component: Description and justification. *Contract*, 39(92-C):0015, 1993.
[13] D. Kirsh. A few thoughts on cognitive overload. *Intellectica*, 1(30):19–51, 2000.
[14] S. Lim and A. Jones. Network anomaly detection system: The state of art of network behaviour analysis. In *International Conference on Convergence and Hybrid Information Technology 2008*, pages 459–465. IEEE, 2008.
[15] Y. Livnat, J. Agutter, S. Moon, R. Erbacher, and S. Foresti. A visualization paradigm for network intrusion detection. In *Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC*, number June, pages 92–99. IEEE, 2005.
[16] P. McLachlan, T. Munzner, E. Koutsofios, and S. North. LiveRAC: interactive visual exploration of system management time-series data. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1483–1492. ACM, 2008.
[17] R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy*, pages 305–316. IEEE, 2010.
[18] T. Taylor, D. Paterson, J. Glanfield, C. Gates, S. Brooks, and J. McHugh. FloVis: Flow Visualization System. *2009 Cybersecurity Applications & Technology Conference for Homeland Security*, pages 186–198, Mar. 2009.
[19] J. Tukey. *Exploratory data analysis*. Menlo Park, CA: Addison-Wesley, 1977.